

Application Note

Remote control of Barix Barionet 50 and 100 using MXE Matrix Mix Engine's Active HTTP API and the OMNEO Dante OCA network interface

MXE Matrix Mix Engines are equipped with an OMNEO Dante OCA network interface for connecting to other systems, using CAT cables and Ethernet network switches.

|--|

Image 1: MXE rear view

The network interface (*OMNEO Dante OCA*) can be found on the MXE's rear panel. It offers in total three network ports: *CONTROL*, *PRIMARY* and *SECONDARY*.

The three network ports can be configured via SONICUE to run either in Transparent, RSTP or Glitch-Free mode.



Image 2: MXE network interface detail view

Requirements for using MXE Task Engine:

MXE Matrix Mix Engine with firmware version 1.4.3119 (or higher)

SONICUE Sound System Software 1.3.0 (or higher) installed on computer

Barionet 100 with firmware version 3.06 (or higher)

Barionet 50 with firmware version 2.05 (or higher)

Documentation (recommended in addition to this application note)

A detailed description of all features and functions of the products mentioned in this application note can be found on the manufacturer's websites and in the Barionet 50/100 device manual.



1. Barix Barionet 50 and 100 (third party)

Applications

Barix's Barionet 50 and 100 universal, programmable I/O device servers offer the following applications with MXE.

- GPIO extension -> number of GPIOs (General Purpose Inputs and Outputs)
- GPIO extension -> remote location of GPIOs

Programming

The example described in this application note requires no programming of the Barionet 50 / 100 itself, as the integrated Common Gateway Interface (CGI) is used for controlling the Barionet's control outputs.

Requesting the GPIO status requires loading of an application "CGI_output state v1_1" from the Barix website and installing it on the Barionet.

For advanced functions and features, like writing small programs in Barix Control Language (BCL) or micro websites to run on the Barionet, please study the product documentation available on the manufacturer's website.

Installation

Both models have a similar housing and are intended to be installed on a DIN-rail.

Variants

The following Barionet universal, programmable I/O device servers have been successfully tested with MXE:

- Barix Barionet 50
- Barix Barionet 100





Image 3: Barix Barionet 50 (left), Barix Barionet 100 (right)



Ethernet port

Besides the analog or digital inputs and outputs, that are different on each of the models, all models feature an Ethernet port for connection to an Ethernet network.

The Ethernet (Eth) port can be found on the front panel of the Barionet 50/100:



Image 4: Barix Barionet 100 front panel simplified view (Barionet 50 is quite similar)

Basic set up – network connection



Image 5: Barix Barionet I/O device server connection to Dynacord MXE via Ethernet network switch



1.1. Barix Barionet 50 universal, programmable I/O device server

Product image



Image 6: Barix Barionet 50 front/side view

Brief description

The Barionet 50 universal, programmable I/O device server is part of Barix's Barionet Series.

It features the following inputs and outputs:

- 4 x digital input
- 4 x relay output
- 1 x Ethernet Interface 10/100 Mbit/s
- 1 x 1-Wire® interface
- o 1 x RS-232 Serial Interface
- 1 x RS-422/485 Serial Interface

Application examples

- Use a hardware switch or relay contact connected to a Barionet 50 digital input to control an MXE DSP mute or set a fault flag.
- Use an LED (with external power supply) connected to a Barionet 50 relay output to signalize a fault in the MXE.
- Use a Barionet 50 relay output to switch smaller loads directly.
- Use a larger relay connected to a Barionet 50 relay output to switch larger currents or voltages potential-free.



1.2. Barix Barionet 100 universal, programmable I/O device server

Product image



Image 7: Barix Barionet 100 front/side view

Brief description

The Barionet 100 universal, programmable I/O device server is part of Barix's Barionet Series.

It features the following inputs and outputs:

- 4 x analog/digital input
- 4 x digital input
- 4 x digital output
- 2 x relay output (max 250 V, 5 A)
- 1 x Ethernet Interface 10/100 Mbit/s
- 1 x 1-Wire® interface
- 1 x RS-232 Serial Interface
- 1 x RS-422/485 Serial Interface

Application examples

- Use a hardware switch or relay contact connected to a Barionet 100 digital input to control an MXE DSP mute or set a fault flag.
- Use an LED (with external power supply) connected to a Barionet 100 digital or relay output to signalize a fault in the MXE.
- Use a Barionet 100 digital output to switch smaller loads directly or via external relays.
- Use a Barionet 100 relay output to switch even larger loads (max 250V, 5A) directly.



2. Barix Barionet 50/100 Common Gateway Interface (CGI)

The Barionet 50/100's CGI serves as a simple and easy to integrate interface to other systems.

The CGI for controlling Barionet 50/100's outputs is enabled by default. No activation is necessary.

General format of rc.cgi command:

http://<barionet IP address>/rc.cgi?o=<I/O address>,<value>[&L=<response page>]

Barionet 50/100 options for the <value> parameter:

Value	Function
0	Set the output to inactive (off)
1	Set the output to active (on)
999	Toggle the output. If it was on, change it to off and vice versa.
2 - 998 or	Toggle the output for n * 100 ms. (e.g. 50 = toggle the output for 5 seconds)
1000 - 9999	

Table 1: Barionet 50/100 options for the <value> parameter

Barionet 50 <I/O address> overview of the relay outputs:

I/O Address	Function
1	Relay 1
2	Relay 2
3	Relay 3
4	Relay 4

Table 2: Barionet 50 adresses of the output relays

Barionet 100 <I/O address> overview of the output relays and digital outputs:

I/O Address	Read/Write	Function
1	Read/Write	Relay 1
2	Read/Write	Relay 2
101	Read/Write	Digital Output 1
102	Read/Write	Digital Output 2
103	Read/Write	Digital Output 3
104	Read/Write	Digital Output 4

Table 3: Barionet 100 adresses of the output relays and digital outputs

Assuming the Barionet 50/100 has an IP address of 192.168.178.98 (as used for testing), the output relays can be controlled with the following commands:

• Example to activate relay 1:

http://192.168.178.98/rc.cgi?o=1,1



• Example to deactivate relay 2:

http://192.168.178.98/rc.cgi?o=2,0

To request the status of all Barionet inputs, outputs or variables, the installation of an additional small BCL application "CGI_output state v1_1" is necessary.

This BCL example application can be downloaded from the Barix website.

To upload the BCL application to the Barionet use the "Advanced update" (WEB) for the Barionet 50 or the TFTP upload for the Barionet 100, see more details in the Barionet manual.

Device specific tables with all I/O addresses of Barionet 50 or 100 can also be found in the Barionet 50/100 manual.

After the installation the status of inputs, outputs and virtual I/O bits can be requested with the following command:

General format of basic.cgi command:

http://<barionet IP address>/basic.cgi?state=<I/O address>

Assuming the Barionet 50/100 has an IP address of 192.168.178.98 (as used for testing), the status of inputs, outputs and virtual I/O bits can be requested with the following commands:

• Example to request digital input 1 (I/O address = 201) status:

http://192.168.178.98/basic.cgi?state=201

• Example to request relay 3 (I/O address = 3) status:

http://192.168.178.98/basic.cgi?state=3

3. Barix Barionet Discovery Tool

The Barionet Discovery Tool can be downloaded from the Barix website. It is very useful to find out for example a Barionet 50/100's IP address and current firmware.

Discovery Tool					ľ	<u></u>		- d X
File					и	8		
Set reply	Ethernet address	IP address	HW type	IPAM type	Product ID	FW version		DHCP name
	00-08-e1-01-3f-2d	192.168.178.98	Barionet 50	generic	Barionet	V02.05		
Device selected:						Get Set	Clear	Web Browser

Image 8: Barix Barionet Discovery Tool



4. Barix Barionet Web Interface

The Barionet 50/100 offer a web interface for monitoring the status of inputs and outputs and configuration of for example the network settings.

Besides that, also firmware updating or loading of BCL programs is done through the web interface.



Image 10: Barix Barionet 50 web interface - SETTINGS page



5. MXE Task Engine with active HTTP API

MXE Task Engine offers *HTTP Protocol* blocks for actively sending HTTP GET and POST requests to control 3rd party products or systems with HTTP control protocol.

5.1. MXE Task Engine configuration for controlling relay outputs on a Barix Barionet 100 and polling the status of these relays.

Hint: When online with the MXE Matrix, the Task Engine shows the status of logic or analog values on the connecting lines between blocks (screenshot: true/false).

Matrix1 - TaskEngine - SONICUE				-	- 🗆 X
≡ File Matrix1		Logic λ	Analog λ		Online
Workspaces Credentials					
Q Search	Main Workspace +				
Analog Operations					_
Logic Operations	MXE Mat	rix	false	HTTP Logic Set	- 81
Text Operations	Power HTTP Logic Get Viogic 1			In HTTPLovic Set	_
Constants	Out false false		false true true	• In	- 61
Scheduler	HTTP Logic Get VLogic 3	VLogic 2 O VLogic 3 O			
Thru	Out •	VLogic 4 🔍			
Utilities					
MXF Matrix					
TDY Amplifier					
					<
PROMATRIX Controller					
	Network: 10.0				
	Memory: 1.8				

Image 11: MXE Task Engine configuration for controlling and polling the status of Barix Barionet 100 relays.

Hint: After creating a new or modifying an existing Task Engine configuration, don't forget to click the *DEPLOY* button when *Online* with the system!

Matrix1 - TaskEngine - SC	DNICUE							-		×
≡ File Matrix1					Logic λ	Analog λ	DEPLOY	k	• <u>-</u> 0	nline
Workspaces	Credentials									
Q Search		Main Workspace	+							_

Image 12: MXE Task Engine DEPLOY button



5.2. MXE Task Engine *MXE Matrix* block configuration for using Virtual Logic and Virtual Analog values as interfaces.

The *MXE Matrix* block, added from the *MXE Matrix* menu, can be modified by selecting it, and then clicking the arrow button on the right-hand side of the worksheet.

- SOI Matrix1 - TaskEngine - SOI	NICUE						-	- 🗆	×
≡ File Matrix1				I	Logic λ	Analog λ	DEPLOY		nline
Workspaces	Credentials								
Q Search		Main Workspace							
Analog Operations									
Logic Operations				MXE Matrix			HTTP Logic Set		
Lugic Operations		_		Power	Power 🔍				
Text Operations		HTTP Logic Get		VLogic 1	Error •		HTTP Logic Set		
Constants			Out 🔍	VLogic 2	VLogic 1				
Scheduler		HTTP Logic Get		 VLogic 5 VLogic 4 	VLogic 2 • VLogic 3 •				
Thru			Out •	L	VLogic 4 ●				
Utilities									
MXE Matrix									
MXE Log MXE Matrix MXE Time							Toggle property	/ pane	- - J

Image 13: MXE Task Engine MXE Matrix block, added from the MXE Matrix menu

The VLogic 1-4 connectors on the MXE Matrix block are configured by selecting Visible Virtual Logics 1-4 in the drop-down menu.

Matrix1 - TaskEngine - SONICUE							-		×
≡ File Matrix1				Logic λ	Analog λ	DEPLOY]- Onl	ine
Workspaces Credentials									
Q Search	Main Workspace +								
Analog Operations	MXE Matrix		MXE-f56130oca	tcp.local.		Select Device			
Logic Operations	Power Power		Visible GPOs						
Text Operations	VLogic 1 Error 0 VLogic 1 VLogic 1		Select visible GPOs						
Constants	 VLogic 2 VLogic 2 VLogic 2 		Visible Digital GPIs						
Scheduler	VLogic 4 VLogic 3 VLogic 4		Select visible Digita	l GPIs					
Thru			Visible Analog GPIs						
Utilities			Select visible Analo	g GPIs					
MXE Matrix			Visible Virtual Logics						_
IPX Amplifier		>	4 × Select visibl	le Virtual Log	ics				
HTTP Protocol			✓ 1						
OCA Protocol			2						
PROMATRIX Controller			¥ 2						
			✓ 3						
			✓ 4						

Image 14: MXE Task Engine MXE Matrix block configuration with visible Virtual Logics 1-4



It's important to select the *Device* with the *Select Device* drop-down menu (therefore the Devices must be visible on network!).

Matrix1 - TaskEngine - SONICUE			- 🗆 X
≡ File Matrix1		Logic λ Analog	λ DEPLOY →子 Online
Workspaces Credentials			
, Q Search	Main Workspace +		
Analog Operations	MXE Matrix	MVE Matrix	
Logic Operations	Power Power	MAE Mailix	
Text Operations	VLogic 1 Error		
Constants	 VLogic 2 VLogic 3 VLogic 2 	Title	
Scheduler	 VLogic 4 VLogic 3 	MXE Matrix	
These	VLogic 4 •	Credentials	
Thru		undefined	~
Utilities		Device	Select Device
MXE Matrix		MXE-f56130, oca. tcp.local.	Select Device
IPX Amplifier	>		
HTTP Protocol		Visible GPOs	Matrix1 Ju
OCA Protocol		Select visible GPOs	TPC1-2233
		Visible Digital GPIs	
PROMATRIX Controller		Select visible Digital GPIs	WPN1-f7204b

Image 15: MXE Task Engine MXE Matrix block selection of Device

5.3. MXE Task Engine *HTTP Logic Set* blocks added from the *HTTP Protocol* menu

Matrix1 - TaskEngine - SONICUE					-	
≡ File Matrix1			Logic λ	Analog λ	DEPLOY]- Online
Workspaces Credentials						
, Q. Search	Main Workspace +					
Analog Operations		MYE Matrix	,		HTTP Logic Sot	
Logic Operations			Dowor D			
Text Operations	HTTP Logic Get	 Power VLogic 1 	Error •		HTTP Logic Set	
Constants	Out •	VLogic 2	VLogic 1 •			
Scheduler	HTTP Logic Get	 VLogic 3 VLogic 4 	VLogic 2 O			
Thru	Out •		VLogic 4 🄍			
Utilities						
MXE Matrix						
IPX Amplifier						<
HTTP Protocol						
HTTP Analog Get						
HTTP Analog Set						
HTTP Logic Get						
HTTP Logic Set						

Image 16: MXE Task Engine HTTP Logic Set blocks, added from the HTTP Protocol menu



The HTTP Logic Set blocks need to be configured with the following information:

- *Title* optional, should be edited for better overview
- *Protocol* HTTP or HTTPS (S = secure)
- Host IP address of the device to be controlled
- *Port* standard = 80 for HTTP, 443 for HTTPS
- Basic Auth Credentials optional, to be used if the device to be controlled requires a login
- *Method* GET or POST, depending on implementation
- URL command to be sent to the device to be controlled
- Body optional, only necessary for method POST
- If input reads true, ... value sent if input reads true, replaces %value% in URL or Body
- If input reads false, ... value sent if input reads false, replaces %value% in URL or Body

In the screenshot example the command */rc.cgi?o=1,%value%* is used to control Relay 1, sending value = 1 (Relay on) or value = 0 (Relay off).

Matrix1 - TaskEngine - SONICUE						- 🗆 ×
≡ File Matrix1			Logic λ	Analog λ	DEPLOY]- Online
Workspaces Credentials						
Q Search	Main Workspace +					
Analog Operations						
Logic Operations		HTTP Logic Set				
Text Operations	HTTP Logic Get	Title				
Constants		HTTP Logic Set				
Scheduler	HTTP Logic Get	Protocol				
Thru		нттр				~
Utilities		Host				
MXE Matrix		192.168.1.101				
IPX Amplifier						
HTTP Protocol		80				- 1 +
HTTP Analog Get						
HTTP Analog Set		Basic Auth Credentials				~
HTTP Logic Get	>					
HTTP Logic Set		Method				
OCA Protocol						
PROMATRIX Controller		URL	od			
		/rc.cgi?o=1,%value	% 9			
		Body (only used by meth	od POST)			_
		Body (only used by	method POST			
		If input reads true, replac	ce %value% wi	th		
		1				
		If input reads false, repla	ace %value% w	ith		
		0				

Image 17: MXE Task Engine HTTP Logic Set blocks information to be configured



Matrix1 - TaskEngine - SONICUE				-		×
≡ File Matrix1		Logic λ	Analog λ	DEPLOY	} Or	line
Workspaces Credential	5					
Q Search	Main Workspace +					
Analog Operations Logic Operations Text Operations Constants Scheduler Thru Utilities	HTTP Logic Get Out • HTTP Logic Get Out •	MXE Matrix Power Power VLogic 1 Error VLogic 2 VLogic 1 VLogic 3 VLogic 2 VLogic 4 VLogic 3 VLogic 4 VLogic 4		HTTP Logic Set In HTTP Logic Set In		
IPX Amplifier						<
HTTP Protocol HTTP Analog Get HTTP Analog Set HTTP Logic Get HTTP Logic Set						

5.4. MXE Task Engine *HTTP Logic Get* block added from the HTTP Protocol menu

Image 18: MXE Task Engine HTTP Logic Get blocks, added from the HTTP Protocol menu

The HTTP Logic Get blocks need to be configured with the following information:

- *Title*, *Protocol*, *Host*, *Port*, *Basic Auth Credentials*, *Method*, and *URL* (in the example "/basic.cgi?state=1" to request the status of relay 1 of the Barix Barionet 100) as described for the HTTP Logic Set block
- Set output to true if response includes = value check of the response (example ">1<")

Text Operations	HTTP Logic Get	
Constants	Ou	HTTP Logic Get
Scheduler	HTTP Logic Get	
Thru	Ou	
Utilities		
MXE Matrix		Host
IPX Amplifier		192.168.1.101
HTTP Protocol		Port
HTTP Analog Get		80 - +
HTTP Analog Set		Basic Auth Credentials
- HTTP Logic Get	>	undefined ~
HTTP Logic Set		Method
OCA Protocol		GET ~
		URL
		/basic.cgi?state=1
		Set output to true if response includes
		>1<
HTTP Logic Get HTTP Logic Set OCA Protocol PROMATRIX Controller	>	undefined ~ Method

Image 19: MXE Task Engine HTTP Logic Get block Set output to true if response includes configuration



5.5. <u>Alternative</u> MXE Task Engine configuration for polling the status of digital inputs 1-4 of a Barix Barionet 100 and controlling a *SONICUE Expression* (Mute) in MXE DSP.

Matrix1 - TaskEngine - SONICUE	- 0	×
≡ File Matrix1	Logic A Analog A DEPLOY	ine
Workspaces Credentials		
Q Search	Main Workspace +	
Analog Operations	HTTP Logic Get MXE Matrix SONICUE Expression	
Logic Operations Text Operations	Out • false • Power • Power • Mute • Mute • • VLogic 1 Error •	
Constants	Out • false • VLogic 2 VLogic 1 • HTTP Logic Get • vLogic 3 VLogic 2 •	
Scheduler Thru	Out • False VLogic 4 • VLogic 4 •	
Utilities	Out •	
MXE Matrix		
IPX Amplifier		<
HTTP Protocol		
OCA Protocol		
PROMATRIX Controller		

The *HTTP Logic Get* blocks were added as described in the previous chapters.

Image 20: MXE Task Engine configuration for polling the status of digital inputs 1-4 of a Barix Barionet 100 and controlling a Mute in MXE DSP

The HTTP Logic Get blocks need to be configured with the following information:

- *Title*, *Protocol*, *Host*, *Port*, *Basic Auth Credentials*, *Method*, and *URL* (in the example "/basic.cgi?state=201" to request the status of digital input 1 of the Barix Barionet 100)
- Set output to true if response includes = value check of the response (example ">1<")

Q Search	Main Workspace +	
Analog Operations		HTTP Logic Get
Logic Operations	HTTP Logic Get	Protocol
Text Operations	HTTP Logic Get	HTTP ~
Constants		Host
Scheduler	HTTP Logic Get	192.168.1.101
Thru	HTTP Logic Get	Port
Utilities		80 - +
MXE Matrix		Basic Auth Credentials
IPX Amplifier	>	undefined ~
HTTP Protocol		Method
OCA Protocol		GET ~
PROMATRIX Controller		URL
		/basic.cgi?state=201
		Set output to true if response includes
		>1<

Image 21: MXE Task Engine HTTP Logic Get block Set output to true if response includes configuration



5.6. Adding DSP expressions to an MXE Task Engine configuration

With a DSP flyout open, like the Level flyout in the screenshot below, a DSP expression can be easily added to a Task Engine structure via **drag&drop (+ CTRL key pressed)**.

🗾 Matrix1 - TaskEngine - SONICUE		- 🗆 X
≡ File Matrix1	Logic λ Anal	ogλ DEPLOY → Online
Workspaces Credentials		
Q Search	Main Workspace +	
Analog Operations Logic Operations Text Operations Constants Scheduler Thru Utilities MXE Matrix IPX Amplifier HTTP Protocol OCA Protocol PROMATRIX Controller	 HTTP Logic Get HTTP Logic Get Out • 	Level X * * * * * * * * * * * * *

Image 22: MXE Task Engine structure with DSP Level flyout open, structure shown before drag&drop

A SONICUE Expression block with Mute function has been added to the example Task Engine structure via drag&drop (+ CTRL key pressed) from the Level flyout.

Matrix1 - TaskEngine - SONICUE			-	o ×
≡ File Matrix1		Logic λ Analog λ	DEPLOY	-]- Online
Workspaces Credentials				
Q Search	Main Workspace +			
Analog Operations Logic Operations Text Operations Constants Scheduler Thru Utilities MXE Matrix IPX Amplifier HTTP Protocol	 HTTP Logic Get HTTP Logic Get HTTP Logic Get HTTP Logic Get Ulogic 4 Vlogic 4 Ulogic 4 Ulogic 4 Ulogic 4 	Matrix Power • Error • VLogic 1 • VLogic 2 • VLogic 3 • VLogic 4 •	Level ssion Mute •	×
OCA Protocol PROMATRIX Controller				

Image 23: MXE Task Engine structure with SONICUE Expression block added via drag&drop from a DSP flyout



Finally, the SONICUE Expression block (for Mute) needs to be connected to the Task Engine structure, for example to the VLogic 1 connector of the MXE Matrix block.

With this Task Engine structure a Mute in MXE DSP can be controlled from digital input 1 of a Barix Barionet 100 connected somewhere on the network.

🗾 Matrix1 - TaskEngine - SONIC	CUE			- 0	×
≡ File Matrix1			Logic λ Analog	λ DEPLOY	Online
Workspaces (Credentials				
Q Search	Main Workspace +				
Analog Operations	HTTP Logic Get	MXE Matrix		SONICUE Expression	
Text Operations	Ou	et Power VLogic 1	Power ● Error ●	● Mute ●	
Constants	Ou	• VLogic 2 • VLogic 3 • VLogic 4	VLogic 1 • VLogic 2 • VLogic 3 •	7	
Thru	Ou	it •	VLogic 4 ●		
Utilities	Ou	it •			
MXE Matrix					
IPX Amplifier					<
HTTP Protocol					
OCA Protocol					

Image 24: MXE Task Engine structure with SONICUE Expression block connected to the VLogic 1 connector of the MXE Matrix block

6. User interface created in SONICUE Panel Designer

Example User Interface for relay control and status indication, created for TPC-1

The Virtual Logic status can be controlled via Toggle buttons (VLogic 1&2) or visualized by an LED control (VLogic 3&4) configured in SONICUE Panel Designer.



Image 25: SONICUE Panel Designer design for TPC-1 (in *Preview* mode)



Third party product disclaimer:

Dynacord does not take responsibility for the warranty, quality, or availability of Barix products. The Barix products contained within this document were tested successfully at the time of publication. However, Dynacord cannot guarantee the compatibility with future models or variations of Barix products, as these may not be compatible. Please refer to the Barix website for product specific information.